WEST Search History

Hide Items | Restore | Clear | Cancel

DATE: Thursday, March 11, 2004

Hide?	Set Name	Query	Hit Count
		DB=PGPB,USPT; PLUR=NO; OP=ADJ	
☐	L29	l23 with L28	27
☐	L28	condition code$1 or predicate$1	18125
☐	L27	l6 and L26	8
☐	L26	l23.ti,ab,clm.	957
☐	L25	l6 and L23	81
☐	L24	l6 same L23	2
☐	L23	(co-processor$1 or coprocessor$1)	7878
☐	L22	l6 and l14	2
☐	L21	L20 not L15	50
☐	L20	(L12 or L13) and L19	58
☐	L19	(712/35).ccls.	180
☐	L18	L17 not L15	75
☐	L17	(L12 or L13) and L16	75
☐	L16	(714/35).ccls.	186
☐	L15	(L12 or L13) and L14	68
☐	L14	(712/34).ccls.	212
☐	L13	(interface$1) with program$5	78307
☐	L12	(interface$1) with configur$4	30420
☐	L11	(L8 or L9) with configur$4	6
☐	L10	(L8 or L9) with program$5	37
☐	L9	co-processor interface	79
☐	L8	coprocessor interface	225
☐	L7	coprocessor with functional units	40
☐	L6	data with L5	2588
☐	L5	L4 adj l3	24003
☐	L4	out	2520860
☐	L3	order	2379551
☐	L2	out of order	0
☐	L1	"out of order"	0

END OF SEARCH HISTORY

First Hit Fwd Refs

End of Result Set

☐ Generate Collection | Print

L22: Entry 2 of 2 File: USPT Apr 18, 2000

US-PAT-NO: 6052771
DOCUMENT-IDENTIFIER: US 6052771 A

TITLE: Microprocessor with pipeline synchronization

DATE-ISSUED: April 18, 2000

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Heller, Jr.; Thomas J.	Rhinebeck	NY		
Boyd; William Todd	Poughkeepsie	NY		

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
International Business Machines Corporation	Armonk	NY			02

APPL-NO: 09/ 008792 [PALM]
DATE FILED: January 20, 1998

INT-CL: [07] G06 F 15/16

US-CL-ISSUED: 712/34
US-CL-CURRENT: 712/34

FIELD-OF-SEARCH: 712/23, 712/1, 712/14, 712/31, 712/34, 709/208, 709/400

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

Search Selected | Search ALL | Clear

	PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
☐	4574349	March 1986	Rechtschaffen	711/154
☐	4901233	February 1990	Liptay	712/218
☐	5577200	November 1996	Abramson et al.	712/200
☐	5752035	May 1998	Trimberger	712/229
☐	5909565	June 1999	Morikawa et al.	712/200
☐	5923892	July 1999	Levy	712/31

ART-UNIT: 273

PRIMARY-EXAMINER: Eng; David Y.

ATTY-AGENT-FIRM: Augspurger; Lynn L.

ABSTRACT:

A system and method for improving microprocessor computer system out of order support via register management with synchronization of multiple pipelines and providing for processing a sequential stream of instructions in a computer system having a first and a second processing element, each of the processing elements having its own state determined by a setting of its own general purpose and control registers. When at any point in the processing of the sequential stream of instructions by the first processing element it becomes beneficial to have the second processing element begin continued processing of the same sequential instruction stream then the first and second processing elements process the sequential stream of instructions and may be executing the very same instruction but only one of said processing elements is permitted to change the overall architectural state of the computer system which is determined by a combination of the states of the first and second processing elements. The second processor will have more pipeline stages than the first in order processor to feed the first processor and reduce the finite cache penalty and increase performance. The processing and storage of results of the second processor does not change the architectural state of the computer system. Results are stored in its gprs or its personal storage buffer. Resynchronization of states with a coprocessor occurs upon an invalid op, a stall or a computed specific benefit to processing with the coprocessor as a speculative coprocessor.

27 Claims, 8 Drawing figures

First Hit Fwd Refs

End of Result Set

Generate Collection Print

L22: Entry 2 of 2 File: USPT Apr 18, 2000

DOCUMENT-IDENTIFIER: US 6052771 A
TITLE: Microprocessor with pipeline synchronization

Detailed Description Text (28):
Just after the SFE enters the Running State, it will begin to execute out of order instructions while the uPCore continues execution at its own pace fetching its instructions from the cache 203 including those the speculative engine processing element SFEs storage references that are supplied to cache storage 203 of instruction and operand data in advance of their use by the uPCore conventional processing element. In the preferred embodiment the uPCore can be excluisively designed as an in order processor, or one optomized for in order processing or be one able to handle processing instructions of instructions where substantially less than 95% of all instructions will not benefit from prediction. Therefore it may experience a pipeline stall in the case of a L1 cache miss. The SFE may continue past the instruction which caused the stall since it is capable of out of order execution. During the time that the SFE is running it generates fetch references that are sent to both the Instruction and Data Caches over interface 207 and to the Store Buffers over interface 208. A cache miss is experienced if both the Caches and the Store Buffers do not have the desired data. Instructions and Operands are returned to the SFE over interface 207 in the case where there is no relevant entry in the Store Buffer, and are returned over interface 208 if there is a relevant entry in the Store Buffer. SFE store references are not sent to the Instruction and Data Caches and are instead sent to the Store Buffer. In this way, the results of SFE store instructions can be made available to subsequent instructions executed on the SFE without altering the architectural state of the uPCore and Caches. All SFE stores are kept in the Store Buffers.

Current US Original Classification (1):
712/34

First Hit Fwd Refs

Generate Collection Print

L24: Entry 1 of 2 File: USPT Jun 11, 2002

US-PAT-NO: 6405267
DOCUMENT-IDENTIFIER: US 6405267 B1

TITLE: Command reordering for out of order bus transfer

DATE-ISSUED: June 11, 2002

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Zhao; Randy X.	Fremont	CA		
Ho; Chien-Te	Sunnyvale	CA		
Fong; Steve	Milpitas	CA		

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
S3 Graphics Co., Ltd.				KY	03

APPL-NO: 09/ 236062 [PALM]
DATE FILED: January 22, 1999

INT-CL: [07] G06 F 13/00, G06 F 13/28, G06 F 13/38, G06 F 9/30, G06 F 12/00

US-CL-ISSUED: 710/35; 710/5, 710/20, 710/21, 710/33, 710/38, 710/39, 710/52, 710/129, 345/537, 345/511, 345/520, 345/522, 345/526, 345/538, 711/202, 711/217, 711/218, 712/206, 712/215
US-CL-CURRENT: 710/35; 345/520, 345/522, 345/530, 345/537, 345/538, 710/20, 710/21, 710/305, 710/33, 710/38, 710/39, 710/5, 710/52, 711/202, 711/217, 711/218, 712/206, 712/215

FIELD-OF-SEARCH: 709/213, 709/215, 709/238, 709/239, 709/240, 709/245, 710/1, 710/3, 710/4, 710/5, 710/7, 710/20, 710/21, 710/29, 710/33, 710/35, 710/36, 710/38-40, 710/52-57, 710/61-64, 710/72, 710/74, 710/129, 710/244, 711/1-3, 711/141, 711/145, 711/147, 711/150, 711/151, 711/153-155, 711/202, 711/210, 711/218, 711/217, 711/220, 712/32, 712/34-36, 712/204-208, 712/214-216, 712/225, 345/520, 345/522, 345/537, 345/538, 345/565, 345/566, 345/567, 345/511, 345/526

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

Search Selected Search ALL Clear

	PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
☐	5057998	October 1991	Hirokawa	364/200

☐	5230064	July 1993	Kuo et al.	395/162
☐	5299309	March 1994	Kuo et al.	395/162
☐	5321836	June 1994	Crawford et al.	711/206
☐	5388207	February 1995	Chia et al.	395/164
☐	5432912	July 1995	Kihara	395/275
☐	5561780	October 1996	Glew et al.	711/126
☐	5612863	March 1997	De Lange	364/134
☐	5630075	May 1997	Joshi et al.	395/250
☐	5664148	September 1997	Mulla et al.	711/138
☐	5805930	September 1998	Rosenthal et al.	710/57
☐	5852451	December 1998	Cox et al.	345/509
☐	5893155	April 1999	Cheriton	711/144
☐	5905509	May 1999	Jones et al.	345/520
☐	5920881	July 1999	Porterfield	711/2
☐	5945997	August 1999	Zhao et al.	345/430
☐	5964859	October 1999	Steinbach et al.	710/129
☐	6046752	April 2000	Kirkland et al.	345/505
☐	6073210	June 2000	Palanca et al.	711/118
☐	6085263	July 2000	Sharma et al.	710/56
☐	6112262	August 2000	Goettsch	710/35
☐	6122715	September 2000	Palanca et al.	711/154
☐	6157397	December 2000	Bogin et al.	345/511
☐	6173393	January 2001	Palanca et al.	712/224
☐	6202101	March 2001	Chin et al.	710/5
☐	6205520	March 2001	Palanca et al.	711/138
☐	6223258	April 2001	Palanca et al.	711/138
☐	6247102	June 2001	Chin et al.	711/150
☐	6249853	June 2001	Porterfield	711/206
☐	6272600	August 2001	Talbot et al.	711/140
☐	6275243	August 2001	Priem et al.	345/568
☐	6286090	September 2001	Steely, Jr. et al.	711/207
☐	6300953	October 2001	Rivard et al.	345/430
☐	6313845	November 2001	Terry et al.	345/537
☐	6314500	November 2001	Rose	711/148
☐	6329985	December 2001	Tamer et al.	345/356

FOREIGN PATENT DOCUMENTS

FOREIGN-PAT-NO	PUBN-DATE	COUNTRY	US-CL
WO 97/14101	April 1997	WO	

OTHER PUBLICATIONS

"Accelerating Your Graphics--A Diamond Multimedia White Paper", Diamond Multimedia Systems, Inc., 2 pages, 1997.
"Accelerated Graphics Port Interface Specification", Intel Corporation, Rev. 2.0, pp. 79-87, May, 1998.
Datasheet, Intel 440BX AGPset: 82443BX Host Bridge/Controller, Order No. 290633-001, Intel Corporation, Apr., 1998.
Diamond Multimedia Exotic Memory Whitepaper, Diamond Multimedia Systems, Inc., 8 pages, 1997.
"Intel Platforms for Visual Computing: A White Paper on Intel's Visual Computer Initiative", Intel Corporation, 8 pages, Mar., 1997.
S3's Savage3D Accelerator: A Comprehensive Architecture for Superior PC Video, Savage3D.white papers, 9 pages, S3 Incorporated, 1998.
The Accelerated Graphics Port (AGP)--A Diamond Multimedia White Paper, Diamond Multimedia Systems, Inc., 2 pages, 1997.
"Write Combining Memory Implementation Guidelines", Intel Corporation, 17 pgs., Nov. 1998.

ART-UNIT: 2182

PRIMARY-EXAMINER: Gaffin; Jeffrey

ASSISTANT-EXAMINER: Nguyen; Tanh Q

ATTY-AGENT-FIRM: Carr & Ferrell LLP

ABSTRACT:

A system and method for increasing effective bus bandwidth in communicating with a graphics device. Graphics commands and associated parameters are written into a contiguous region of system memory and transmitted in a weakly ordered fashion over a bus to a graphics device. The graphics device reorders the incoming data into the same order as which the data was written into the contiguous region of system memory, thereby allowing the use of order dependent encoded commands with the weakly ordered bus interface.

16 Claims, 15 Drawing figures

L24: Entry 1 of 2 File: USPT Jun 11, 2002

DOCUMENT-IDENTIFIER: US 6405267 B1
TITLE: Command reordering for out of order bus transfer

Detailed Description Text (2):
The present invention provides a system and a method of increasing effective bus bandwidth utilization by reordering out of order data received from a weakly ordered bus interface. FIG. 1 illustrates a computer system using the present invention. The computer system includes a CPU 1 which communicates with a graphics device over a bus 7. A graphics driver program runs on the CPU. The graphics driver creates drawing commands and parameters utilized in conjunction with the drawing commands. The driver arranges the drawing commands and parameters into 32 bit registers. These 32 bit registers are reserved regions of physical memory address space, and in the embodiment of FIG. 1 are a contiguous region A of system memory 3 or are sequentially ordered areas of memory identifiable by address. In the embodiment of FIG. 1 the contiguous region A is defined as a write combining memory type. Further, the graphics driver sequentially writes commands and parameters into sequentially increasing address locations of the memory, until an upper bound is reached. Once the upper bound is reached, the graphics driver begins writing data, i.e., commands and parameters, starting at the lower bound when a command is ready to be written. Using the bus, the commands and parameters are communicated to the graphics device 5, which often includes a co-processor.

First Hit Fwd Refs

End of Result Set

Generate Collection Print

L24: Entry 2 of 2 File: USPT Apr 4, 2000

US-PAT-NO: 6047367
DOCUMENT-IDENTIFIER: US 6047367 A

TITLE: Microprocessor with improved out of order support

DATE-ISSUED: April 4, 2000

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Heller, Jr.; Thomas J.	Rhinebeck	NY		

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
International Business Machines Corporation	Armonk	NY			02

APPL-NO: 09/ 009828 [PALM]
DATE FILED: January 20, 1998

INT-CL: [07] G06 F 9/38

US-CL-ISSUED: 712/23; 712/215
US-CL-CURRENT: 712/23; 712/215

FIELD-OF-SEARCH: 712/200, 712/23, 712/215

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

Search Selected Search ALL Clear

	PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
☐	4574349	March 1986	Rechtschaffen	711/154
☐	4901233	February 1990	Liptay	712/218
☐	5559976	September 1996	Song	395/375
☐	5751981	May 1998	Witt	395/380

ART-UNIT: 273

PRIMARY-EXAMINER: Coleman; Eric

ATTY-AGENT-FIRM: Augspurger; Lynn L.

ABSTRACT:

A system and method for improving microprocessor computer system out of order support via register management with synchronization of multiple pipelines and providing for processing a sequential stream of instructions in a computer system having a first and a second processing element, each of the processing elements having its own state determined by a setting of its own general purpose and control registers. When at any point in the processing of said sequential stream of instructions by said first processing element it becomes beneficial to have the second processing element begin continued processing of the same sequential instruction stream then the first and second processing elements process the sequential stream of instructions and may be executing the very same instruction but only one of said processing elements is permitted to change the overall architectural state of said computer system which is determined by a combination of the states of said first and second processing elements. The second processor will have more pipeline stages than the first in order processor to feed the first processor and reduce the finite cache penalty and increase performance. The processing and storage of results of the second processor does not change the architectural state of the computer system. Results are stored in its gprs or its personal storage buffer. Resynchronization of states with a coprocessor occurs upon an invalid op, a stall or a computed specific benefit to processing with the coprocessor as a speculative coprocessor.

11 Claims, 8 Drawing figures

First Hit Fwd Refs

End of Result Set

Generate Collection Print

L24: Entry 2 of 2 File: USPT Apr 4, 2000

DOCUMENT-IDENTIFIER: US 6047367 A
TITLE: Microprocessor with improved out of order support

CLAIMS:

1. A computer system having a hierarchical memory with cache storage for instructions and data, comprising,

at least one conventional processing element for processing instructions with at least one instruction pipeline of a defined length and defined delay per pipeline stage;

and an additional speculative engine processing element for processing instructions, including out-of-order instructions, for instruction sequences which derive finite cache improvement from out-of-order processing, wherein

said additional speculative engine processing element and said conventional processing element are coupled for action in concert to process a stream of instructions with said conventional processing element maintaining the architectural state of concerted action with its pipeline handles while said speculative engine processing element handles speculative processes whose result may not change the architectural state of the computer system but which improves the finite cache penalty seen by said conventional processing element, and wherein said speculative processing element includes

m registers in a central processing area, m being greater than a predetermined n number of instruction addressable GPRs identified by one or more binary fields of an instruction, and provision for out of order instruction execution and for processing a conditional branch instruction based on a branch direction guess,

said conventional processing element being coupled to a coprocessor providing said additional speculative engine processing element for independent prefetching and execution of instructions to improve the sequence of storage references as seen by said conventional processing element,

said system using a register management process enabling generation of speculative memory references to said storage hierarchy including said instruction and data cache coupled to and shared by both said additional speculative engine processing element and said conventional processing element,

and further including,

an instruction and data cache coupled to both said conventional out of order processing element and to said coprocessor providing said additional speculative engine processing element for fetching instructions and data, and an independent store buffer for bidirectional transfer of instructions and data in response to store and fetch commands of said additional speculative engine processing element.

11. A microprocessor having a hierarchical memory with cache storage for instructions and data, comprising,

at least one conventional processing element for processing instructions with multiple pipelines of a defined length and defined delay per pipeline stage;

and an additional speculative engine processing element for processing instructions, including out-of-order instructions, for instruction sequences which derive finite cache support from out-of-order processing, wherein

said additional speculative engine processing element and said conventional processing element are coupled for action in concert to process a stream of instructions with said conventional processing element maintaining the architectural state of concerted action with its pipeline handles while said speculative engine processing element handles speculative processes whose result may not change the architectural state of the computer system but which improves the finite cache penalty seen by said conventional processing element with register management and synchronization of said multiple pipelines, and further including,

an instruction and data cache coupled to both said conventional out of order processing element and to said coprocessor providing said additional speculative engine processing element for fetching instructions and data, and an independent store buffer for bidirectional transfer of instructions and data in response to store and fetch commands of said additional speculative engine processing element.

First Hit Fwd Refs

Generate Collection Print

L29: Entry 8 of 27 File: USPT Aug 28, 2001

US-PAT-NO: 6282633
DOCUMENT-IDENTIFIER: US 6282633 B1

TITLE: High data density RISC processor

DATE-ISSUED: August 28, 2001

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Killian; Earl A.	Los Altos Hills	CA		
Gonzalez; Ricardo E.	Menlo Park	CA		
Dixit; Ashish B.	Mountain View	CA		
Lam; Monica	Menlo Park	CA		
Lichtenstein; Walter D.	Belmont	MA		
Rowen; Christopher	Santa Cruz	CA		
Ruttenberg; John C.	Newton	MA		
Wilson; Robert P.	Palo Alto	CA		

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
Tensilica, Inc.	Santa Clara	CA			02

APPL-NO: 09/ 192395 [PALM]
DATE FILED: November 13, 1998

INT-CL: [07] G06 K 9/30

US-CL-ISSUED: 712/208; 712/210, 712/226
US-CL-CURRENT: 712/208; 712/210, 712/226

FIELD-OF-SEARCH: 712/24, 712/41, 712/200, 712/208, 712/210, 712/223, 712/225, 712/226

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

Search Selected Search ALL Clear

	PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
☐	5155820	October 1992	Gibson	712/210
☐	5426743	June 1995	Phillips et al.	712/221

☐	5581717	December 1996	Boggs et al.	712/208
☐	5638524	June 1997	Kiuchi et al.	712/221
☐	5991870	November 1999	Koumura et al.	712/208

FOREIGN PATENT DOCUMENTS

FOREIGN-PAT-NO	PUBN-DATE	COUNTRY	US-CL
0 363 222 A2	April 1990	EP	
0 374 419 A2	June 1990	EP	
0 660 223 A2	June 1995	EP	
0 696 772 A2	February 1996	EP	
0 924 601 A2	June 1999	EP	
WO 93/01543	January 1993	WO	

OTHER PUBLICATIONS

Texas Instruments, "TMS32010 User's Guide", 1983, p. 3-7.*
Lefugy, C. et al., "improving Code Density using Compression Techniques", 12/97. p. 194-204.*
Conte, T.M. et al., "Instruction Fetch Mechanisms for VLIW Architectures Compressed Encodings", 12/96. p. 201-211.

ART-UNIT: 282

PRIMARY-EXAMINER: Niebling; John F.

ASSISTANT-EXAMINER: Whitmore; Stacy

ATTY-AGENT-FIRM: Pillsbury Winthrop LLP

ABSTRACT:

A RISC processor implements an instruction set which, in addition to optimizing a relationship between the number of instructions required for execution of a program, clock period and average number of clocks per instruction, also is designed to optimize the equation S=IS * BI, where S is the size of program instructions in bits, IS is the static number of instructions required to represent the program (not the number required by an execution) and BI is the average number of bits per instruction. Compared to conventional RISC architectures, this processor lowers both BI and IS with minimal increases in clock period and average number of clocks per instruction. The processor provides good code density in a fixed-length high-performance encoding based on RISC principles, including a general register with load/store architecture. Further, the processor implements a simple variable-length encoding that maintains high performance.

21 Claims, 5 Drawing figures

L29: Entry 8 of 27 File: USPT Aug 28, 2001

DOCUMENT-IDENTIFIER: US 6282633 B1
TITLE: High data density RISC processor

Detailed Description Text (56):
This is a new variant of the condition code-based compare and branch found in many earlier instruction sets, as discussed above. Earlier instruction sets have multiple shared multi-bit condition codes between the processor and its coprocessors (e.g., the PowerPC) and used multiple per-coprocessor single-bit condition codes (e.g., MIPS). The preferred embodiment of the present invention uses multiple shared single-bit condition codes.

First Hit Fwd Refs

Generate Collection Print

L29: Entry 10 of 27 File: USPT Dec 14, 1999

US-PAT-NO: 6003129
DOCUMENT-IDENTIFIER: US 6003129 A

TITLE: System and method for handling interrupt and exception events in an asymmetric multiprocessor architecture

DATE-ISSUED: December 14, 1999

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Song; Seungyeon Peter	Los Altos	CA		
Mohamed; Moataz A.	Santa Clara	CA		
Park; Heon-Chul	Cupertino	CA		
Nguyen; Le	Monte Sereno	CA		

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
Samsung Electronics Company, Ltd.	Seoul			KR	03

APPL-NO: 08/ 699294 [PALM]
DATE FILED: August 19, 1996

PARENT-CASE:
CROSS-REFERENCE TO RELATED APPLICATIONS This patent document is related to and incorporates by reference, in their entirety, the following concurrently filed patent applications: U.S. patent application Ser. No. 08/699,280, entitled "MULTIPROCESSOR OPERATION IN A MULTIMEDIA SIGNAL PROCESSOR", Le Nguyen. U.S. patent application Ser. No. 08/699,597, entitled "SINGLE-INSTRUCTION-MULTIPLE-DATA PROCESSING IN A MULTIMEDIA SIGNAL PROCESSOR", Le Nguyen. U.S. patent application Ser. No. 08/699,280, entitled "EFFICIENT CONTEXT SAVING AND RESTORING IN MULTIPROCESSORS", S. P. Song et al. U.S. patent application Ser. No. 08/699,280, entitled "SYSTEM AND METHOD FOR HANDLING SOFTWARE INTERRUPTS WITH ARGUMENT PASSING", S. P. Song et al. U.S. patent application Ser. No. 08/699,295, entitled "METHODS AND APPARATUS FOR PROCESSING VIDEO DATA", C. Reader et al. U.S. patent application Ser. No. 08/697,086, entitled "SINGLE-INSTRUCTION-MULTIPLE-DATA PROCESSING USING MULTIPLE BANKS OF VECTOR REGISTERS", Le Nguyen et al. U.S. patent application Ser. No. 08/699,585, entitled "SINGLE-INSTRUCTION-MULTIPLE-DATA PROCESSING WITH COMBINED SCALAR/VECTOR OPERATIONS", Le Nguyen et al.

INT-CL: [06] G06 F 17/00

US-CL-ISSUED: 712/244; 395/800.28
US-CL-CURRENT: 712/244; 712/28

FIELD-OF-SEARCH: 395/591, 395/738, 395/739, 395/740, 395/800.3, 395/800.28

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

Search Selected | Search ALL | Clear

	PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
☐	5193158	March 1993	Kinney et al.	395/591
☐	5278647	January 1994	Hingorani et al.	
☐	5319753	June 1994	MacKenna et al.	
☐	5510842	April 1996	Phillips et al.	
☐	5576765	November 1996	Cheney et al.	
☐	5594905	January 1997	Mital	395/733
☐	5668599	September 1997	Cheney et al.	
☐	5729279	March 1998	Fuller	

ART-UNIT: 277

PRIMARY-EXAMINER: Ramirez; Ellis B.

ATTY-AGENT-FIRM: Skjerven, Morrill, MacPherson, Franklin & Friel LLP Koestner; Ken J.

ABSTRACT:

A multiprocessor computer system includes a plurality of processors, called asymmetric processors, having mutually dissimilar control and data-handling characteristics. The asymmetric processors are controlled by a single operating system although the individual processors have instruction sets that are mutually independent of the other processors. The multiprocessor computer system uses a multiprocessor architectural definition of interrupt and exception handling in which a processor, called a data or vector processor, having a large machine state and a large data width detects exceptions but defers interrupt and exception handling operations to another processor, called a control processor, having a small machine state and data width. The small machine state and small data width of the control processor are well suited for executing operating system programs such as interrupt and exception handling since control programs typically involve monitoring and control of individual flags and pointers. The data processor enters an idle state upon reset and when an exception is detected to facilitate system design and programming, and to simplify synchronization of the processors at system reset. A multiprocessor computer system includes a control processor which reads and writes control and status registers within a data processor. The control processor thus controls the operation of the data processor during execution of an operating system or application programs. The control processor has access to the control and status registers of the data processor independent of the data processor execution so that the same control and status registers may be accessed by the control processor and the data processor in parallel.

25 Claims, 9 Drawing figures

L29: Entry 10 of 27 File: USPT Dec 14, 1999

DOCUMENT-IDENTIFIER: US 6003129 A
TITLE: System and method for handling interrupt and exception events in an asymmetric multiprocessor architecture

Detailed Description Text (18):
The control processor 204 starts the vector processor 206 using a STARTVP instruction. The STARTVP instruction causes the vector processor 206 to enter a VP.sub.-- RUN state indicating that the vector processor 206 is executing. STARTVP does nothing if the vector processor 206 is already in the VP.sub.-- RUN state. No result is communicated to the control processor 204 in response to the STARTVP instruction and the control processor 204 continues execution following STARTVP. The STARTVP instruction is invoked following a coprocessor data operations format (CDP) instruction which sets a condition code. The STARTVP instruction is invoked using the following assembler syntax:

Detailed Description Text (20):
The control processor 204 stops the vector processor 206 using an interrupt vector processor (INTVP) instruction. The INTVP instruction causes the vector processor 206 to enter a VP.sub.-- IDLE state indicating that the vector processor 206 is not executing. INTVP does nothing if the vector processor 206 is not executing. No result is communicated to the control processor 204 in response to the INTVP instruction and the control processor 204 continues execution following INTVP. The INTVP instruction is invoked following a coprocessor data operations format (CDP) instruction which sets a condition code. The INTVP instruction is executed only if the condition is true. Thus, the INTVP instruction is invoked using the following assembler syntax:

Detailed Description Text (22):
The control processor 204 tests the operating state of the vector processor 206, including testing for synchronization, using a test and set (TESTSET) instruction. TESTSET reads a user-extended register and sets bit 30 of the register to 1, supplying producer/consumer type synchronization between the vector processor 206 and the control processor 204. TESTSET causes the control processor 204 to stall until the register is transferred. The control processor 204 requests the TESTSET instruction following a coprocessor register transfer operations (MRC,MCR) instruction which sets a condition code. The TESTSET instruction is executed only if the condition is true. The TESTSET instruction is called using the following assembler syntax:

Detailed Description Text (25):
For example, a MFVP instruction moves data from a vector processor scalar/special-purpose register to a general register in the control processor 204. The MFVP instruction is called subsequent to a coprocessor register transfer operations (MRC,MCR) instruction which sets a condition code. The MFVP instruction is executed only if the condition is true. The MFVP instruction is called using the following assembler syntax:

Detailed Description Text (27):
A move to vector processor (MTVP) instruction moves data to a vector processor scalar/special-purpose register from a general register in the control processor

204. The MTVP instruction is called subsequent to a coprocessor register transfer operations (MRC,MCR) instruction which sets a condition code. The MTVP instruction is executed only if the condition is true. The MTVP instruction is called using the following assembler syntax:

Detailed Description Text (31):
A move to extended register (MTER) instruction moves data from the control processor 204 to a specified extended register. The MTER instruction is called subsequent to a coprocessor register transfer operations (MRC,MCR) instruction which sets a condition code. The MTER instruction is executed only if the condition is true. The MTER instruction is called using the following assembler syntax:

Detailed Description Text (33):
A move from extended register (MFER) instruction moves data from an extended register in a designated coprocessor to a register in the control processor 204. The MFER instruction is called subsequent to a coprocessor register transfer operations (MRC,MCR) instruction which sets a condition code. The MFER instruction is executed only if the condition is true. The MFER instruction is called using the following assembler syntax:

First Hit Fwd Refs

Generate Collection Print

L29: Entry 14 of 27 File: USPT Apr 4, 1995

US-PAT-NO: 5404560
DOCUMENT-IDENTIFIER: US 5404560 A

TITLE: Microprocessor having external control store

DATE-ISSUED: April 4, 1995

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Lee; Raymond Y.	Salem	NH		
Bessolo; Jeffrey M.	Groton	MA		
Shah; Vyomesh	Tewksbury	MA		
Vincelette; Scott D.	Bethlehem	PA		
Waldstein; Steven M.	Westford	MA		
Nathan; Jeffrey D.	Andover	MA		
Lang; Steven E.	Lincoln Center	MA		

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
Wang Laboratories, Inc.	Lowell	MA			02

APPL-NO: 08/ 083852 [PALM]
DATE FILED: June 25, 1993

PARENT-CASE:
This is a continuation of application Ser. No. 07/189,853, filed on May 3, 1988, now abandoned.

INT-CL: [06] G06 F 15/00

US-CL-ISSUED: 395/800; 364/DIG.1, 364/262.8, 364/243, 364/247, 364/247.3, 364/262.4, 364/243.7
US-CL-CURRENT: 712/208; 712/42

FIELD-OF-SEARCH: 395/400, 395/425, 395/775, 395/800

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

Search Selected Search ALL Clear

PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL

☐	4307445	December 1981	Tredennick et al.	364/200
☐	4330823	May 1982	Retter	364/200
☐	4367524	January 1983	Budde et al.	364/200
☐	4399505	August 1983	Druke et al.	364/200
☐	4415969	November 1983	Bayliss et al.	364/200
☐	4460972	July 1984	Homan et al.	364/900
☐	4466056	August 1984	Tanahashi	364/200
☐	4495563	January 1985	McDonough	364/200
☐	4513174	April 1985	Herman	364/200
☐	4550369	October 1985	Rokutanda et al.	364/200
☐	4554627	November 1985	Holland et al.	364/200
☐	4569018	February 1986	Hummel et al.	364/200
☐	4648034	March 1987	Heninger	364/200
☐	4677586	June 1987	Magar et al.	364/900
☐	4771376	September 1988	Kamiya	364/200
☐	4794524	December 1988	Carberry et al.	364/200
☐	4811345	March 1989	Johnson	364/200
☐	4825363	April 1989	Baumann et al.	364/200
☐	4835679	May 1989	Kida et al.	364/200
☐	4839795	June 1989	Iwaksaki	364/200
☐	5041969	August 1991	Kawasaki et al.	395/200

OTHER PUBLICATIONS

Vora et al, "A VLSI Implementation of the 32-Bit Eclipse Architecture", Proceedings of IEEE Int'l Conf on Computer Design (ICCD '83) pp. 311-314.
"80286 and 80287 Programmer's Reference Manual", Intel 1987, pp. 2-14-2-16 & 3-20.

ART-UNIT: 232

PRIMARY-EXAMINER: Rudolph; Rebecca L.

ATTY-AGENT-FIRM: Milik; Kenneth L.

ABSTRACT:

A central processing unit (CPU) 10 comprises an external control memory for storing microinstructions which correspond to macroinstructions read from a system memory. The microinstructions are 56 bits in length and are read in 28-bit segments. CPU 10 also comprises an internal memory management unit (MMU) 18 which comprises a plurality of address translation entry (ATE) registers four of which are permanent and sixteen of which are temporary in that the storage of a new translation entry occurs in a least recently used temporary translation entry register. CPU 10 also comprises a plurality of status register bits, some of which are settable only by predefined microinstructions. All of the status register bits are branchable. CPU 10 further comprises a condition code register the state of which may be determined

by input signal pins. CPU 10 also comprises address generation logic which may generate a 24, 31 or 32 bit address upon a 32-bit address bus. The address generation logic is further operable for generating a memory storage address; the data being supplied by external logic, such as a coprocessor.

17 Claims, 21 Drawing figures

L29: Entry 14 of 27 File: USPT Apr 4, 1995

DOCUMENT-IDENTIFIER: US 5404560 A
TITLE: Microprocessor having external control store

Detailed Description Text (18):
In accordance with aspects of the presently preferred embodiment of the invention, coprocessor 3 may be coupled to the output of control memory 14 for directly receiving and decoding the microinstruction stream as it is fetched for execution by CPU 10. The coprocessor 3 may also be coupled, via CPU 10 input signal pins XCC0 and XCC1, to CPU 10 whereby coprocessor 3 is enabled to directly set the contents of a condition code register within CPU 10. These aspects of the invention provide for a tightly coupled, synchronous CPU/coprocessor interface. CPU 10 is also provided with bus request and bus grant signal lines which provide for a loosely coupled, asynchronous CPU/coprocessor interface. Furthermore, the coprocessor 3 may be coupled to the system data bus such that, in accordance with one aspect of the invention, coprocessor 3 may read data from or store data within the system memory 2 at an address provided by CPU 10. This aspect of the invention will be discussed below.

Detailed Description Text (88):
In accordance with one aspect of the invention the condition code type CCX enables the setting of the condition codes in PMR [0:1] from the input signal pins XCC0 and XCC1. This mode of operation is especially advantageous when the CPU 10 is operating in conjunction with the coprocessor 3 in that it permits a tightly coupled interface to be formed between the CPU 10 and the coprocessor. For example, the coprocessor 3, by controlling the state of XCC0 and XCC1, may directly affect the execution of a CPU 10 microinstruction, such as the Branch with Condition Code Check (BCC) which will be described in Table 9.

Detailed Description Text (144):
For systems using MMU 18, each hardware acceleration unit may interface to CPU 10 via the memory data bus and any or all of the five control lines: an input to CPU 10's wait logic, an input to CPU 10's coprocessor trap logic, an input to CPU 10's BNM-time coprocessor trap logic, and two lines to CPU 10's external condition code pins.

Detailed Description Text (145):
For systems having external address translation hardware, each hardware acceleration unit may interface to CPU 10 via the memory data bus and any or all of the four control lines: an input to CPU 10's wait logic, an input to CPU 10's BNM-time coprocessor trap logic, and two lines to CPU 10's external condition code pins.

First Hit Fwd Refs

Generate Collection Print

L29: Entry 18 of 27 File: USPT Feb 5, 1991

US-PAT-NO: 4991080
DOCUMENT-IDENTIFIER: US 4991080 A

TITLE: Pipeline processing apparatus for executing instructions in three streams, including branch stream pre-execution processor for pre-executing conditional branch instructions

DATE-ISSUED: February 5, 1991

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Emma; Philip G.	Danbury	CT		
Pomerene; James H.	Chappaqua	NY		
Rechtschaffen; Rudolph N.	Scarsdale	NY		
Sparacio; Frank J.	North Bergen	NJ		

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
International Business Machines Corporation	Armonk	NY			02

APPL-NO: 06/ 839312 [PALM]
DATE FILED: March 13, 1986

INT-CL: [05] G06F 9/38, G06F 9/28

US-CL-ISSUED: 364/200; 364/263, 364/261.3, 364/261.5, 364/263.1, 364/262.4, 364/261.7
US-CL-CURRENT: 712/206; 712/235, 719/310

FIELD-OF-SEARCH: 364/2MSFile, 364/9MSFile

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

Search Selected Search ALL Clear

	PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
☐	3559183	January 1971	Sussenguth	364/200
☐	3707725	December 1972	Delheim	364/200
☐	4062058	December 1977	Haynes	364/200
☐	4155120	May 1979	Keefer et al.	364/200

☐	4270167	May 1981	Koehler et al.	364/200
☐	4325120	April 1982	Colley et al.	364/200
☐	4338675	July 1982	Palmer et al.	364/748
☐	4442484	April 1984	Childs, Jr. et al.	364/200
☐	4471433	September 1984	Matsumoto et al.	364/200
☐	4477872	October 1984	Losq et al.	364/200
☐	4547849	October 1985	Louie et al.	364/200
☐	4569018	February 1986	Hummel et al.	364/200
☐	4594651	June 1986	Jaswa et al.	364/131
☐	4597041	June 1986	Guyer	364/200
☐	4604691	August 1986	Akagi	364/200
☐	4608659	August 1986	Bradley	364/737
☐	4691277	September 1987	Kronstadt et al.	364/200
☐	4701842	October 1987	Olnowich	364/200
☐	4710866	December 1987	Zolnowsky et al.	364/200
☐	4719570	January 1988	Kawabe	364/200
☐	4725947	February 1988	Shonai	364/200
☐	4742451	May 1988	Bruckert et al.	364/200
☐	4764861	August 1988	Shibuya	364/200
☐	4819154	April 1989	Stiffler	364/200
☐	4821183	April 1989	Hauris	364/200
☐	4827402	May 1989	Wada	364/200
☐	4858104	August 1989	Matsuo	364/200

OTHER PUBLICATIONS

The New Riverside Dictionary by Houghton Mifflin Co., 1984, pp. 640, 641, Riverside Publishing Co.

ART-UNIT: 232

PRIMARY-EXAMINER: Eng; David Y.

ASSISTANT-EXAMINER: Coleman; Eric

ATTY-AGENT-FIRM: Sughrue, Mion, Zinn, Macpeak & Seas

ABSTRACT:

In an approach to reducing delays resulting from resolution of conditional branch instructions, such instructions are pre-executed in a coprocessor which precedes a pipeline processor and prepared an instruction stream for input to the pipeline processor. Because of this pre-execution, the input instruction stream has fewer conditional branches for the pipeline processor to resolve. Also, the coprocessor may handle address generation interlock situations which also cause execution

delays in the pipeline processor.

18 Claims, 17 Drawing figures

L29: Entry 18 of 27 File: USPT Feb 5, 1991

DOCUMENT-IDENTIFIER: US 4991080 A
TITLE: Pipeline processing apparatus for executing instructions in three streams, including branch stream pre-execution processor for pre-executing conditional branch instructions

Brief Summary Text (47):
When a branch is encountered, the branch stream coprocessor pre-executes it, along with the condition code setting instruction which usually immediately precedes it. The comparison between actual actions and targets and predicted actions and targets takes place before instructions are fed into the main stream pipeline. If necessary, the branch history table will be corrected. Correction takes place in this branch stream, rather than in the main stream, so that branch delays during instruction execution are reduced. Also, because the branch stream coprocessor scans instructions two at a time, the coprocessor will tend to stay sufficiently ahead of a main stream processor which actually executes the instructions.

Drawing Description Text (15):
FIG. 14 is a chart showing the procedure for verifying condition codes set by the branch stream coprocessor.

Detailed Description Text (59):
If a Test Under Mask or Compare Logical Immediate instruction is not marked as invalid as described above, it will proceed through the pipeline. The operand will be fetched from cache 101 via the paths 551 and 553, and the pre-execute step 559 will generate the condition code and send it to a BSC (branch stream coprocessor) condition code and valid logic circuit 561. In this logic, an unlikely but possible situation is checked for. A Test Under Mask or Compare Logical Immediate instruction is almost always followed by a Branch on Condition or Branch on Condition Register, in which case the condition code is the proper one for the Branch on Condition or Branch on Condition Register instruction to act on. However, there could be an intervening instruction which could alter the condition code generated by the Test Under Mask or Compare Logical Immediate instruction, thereby rendering the generated condition code unreliable. When such a situation is detected, the condition code is marked invalid The logic for this procedure is shown in tabular form in FIG. 14. Note that three successive instructions enter into this logic.

Detailed Description Text (69):
2. The first instruction in the next instruction pair register 703 is either a Test Under Mask or a Compare Logical Immediate instruction, which only set the condition code. Since the correct condition code value was determined by the branch stream coprocessor 501, and is part of entry in the ready-to-decode buffer 601, it is not necessary to refetch the Test Under Mask or Compare Logical Immediate operand. The condition code value will be carried along through the pipeline of the main stream processor 701 and will be used to update the main stream processor condition code at the proper time during execution of the Test Under Mask or Compare Logical Immediate instruction.